What are the four main components of any general-purpose microprocessor?

- A memory (register): which stores both data and instructions:
- An arithmetic and logic unit (ALU): capable of operating on binary data;
- A control unit: which interprets the instructions in memory and causes them to be executed;
- Input and output (I/O) equipment: operated by the control unit.

At the integrated circuit level, what are the three principal constituents of a computer system?

- Gates,
- Memory cells
- Interconnections among gates and memory cells

Consider a hypothetical 32-bit microprocessor having 32-bit instructions composed of two fields: The first byte contains the opcode and the remainder the immediate operand or an operand address.

a) What is the maximum directly addressable memory capacity (in bytes)?

Address field of the instruction is 24 bits. Therefore, the total memory capacity is
$2^{24} = 2^4 \times 2^{20}$ = 16 Mbytes

b) How many bits are needed for the program counter and the instruction register?

Because the address field of the instruction is 24 bits,

- The program counter must be at least 24 bits.
- If the instruction register is to contain the whole instruction, it will have to be 32-bits long

Consider a 32-bit microprocessor, with a 16-bit external data bus, driven by an 8-MHz input clock. Assume that this microprocessor has a bus cycle whose minimum duration equals four input clock cycles.

What is the maximum data transfer rate that this microprocessor can sustain?

Clock cycle = 1/Clock frequency

Clock cycle = 1/8 MHz = 0.125×10^{-6} s

= 125×10^{-9} s = 125 ns

Bus cycle = number of clocks x Clock cycle

Bus cycle = 4 x 125 ns = 500 ns

2 bytes transferred every 500 ns;

Thus, transfer rate = $2/(500 \times 10^{-9})$

= $2/(5 \times 10^{-7})$

= 0.4×10^{7} = 4 MBytes/sec

An 8-bit microprocessor system has 9 address lines to address relevant memory locations.

Assuming that the data size is 1 byte, what is the address of the last memory location?

Because the processor has 9 address lines the total directly addressable memory size is
2^9 = 512 Bytes.

So, the address of the last memory location will be 2^9-1 = 511

A given microprocessor has words of one byte.

What is the smallest and largest integer that can be represented in the following representation?

The word size is 1 byte (8 bits). So,

Unsigned	0;255
Sign magnitude	-127;127
Ones complement	-127;128
Twos complement	-128;127
Binary coded decimal	00;99

In a signed (2s complement number system) addition, if the “carry in” and the “carry out” of the sign bit differ, there is an overflow.

a) Determine whether there is an overflow in the following operations or not. (use 4 bit 2s complement numbers)

Op.		4+3		-7-6		5+7		-3-2
C.In				0		1		1
		0100		1001		0101		1101
	+	0011	+	1010	+	0111	+	1110
Result		0111		0011		1100		1011
C.Out	0		1		0		1	
O.Flow		no		yes		yes		no

b) Design a circuit that whenever an overflow happens the output becomes 1, otherwise 0.

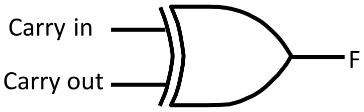
Truth Table

C_i	C_o	F
0	0	0
0	1	1
1	0	1
1	1	0

Overflow Function

$$F=C_i' C_o + C_i C_o' = C_i \oplus C_o$$

Implementation



A computer system has an 8-bit accumulator and 12-bit address bus. If the content of memory location 1F0 is the binary number 11000011;

(a) What is the corresponding decimal value when the number represents an unsigned integer number?

$1\times2^7+1\times2^6+1\times2^1+1\times2^0 = 195$

(b) What is the corresponding decimal value when the number represents a twos complement number?

$-1\times2^7+1\times2^6+1\times2^1+1\times2^0 = -61$

(c) What is the corresponding decimal value when the number represents a ones complement number?

-60

(d) Assuming that the most significant bit is parity bit, what is the corresponding letter when the number represents an ASCII character (hint: ASCII character is represented by 7 bits and 41H represents A)?

If the most significant bit is parity bit, ASCII character represented by 11000011 is 100 0011 (43 in hexadecimal) corresponding to capital letter C.

Consider a dynamic RAM that must be given a refresh cycle 64 times per ms. Each refresh operation requires 150 ns;

What percentage of the memory's total operating time must be given to refreshes?

- In 1 ms, the time devoted to refresh is:

  64 × 150 ns = 9600 ns = 9.6 μs = 0.0096 ms.

- The fraction of time devoted to memory refresh is:

  $(9.6 \times 10^{-6}$ s$)/10^{-3}$ s = 0.0096 = 0.96%.

Following figure shows a simplified timing diagram for a DRAM read operation over a bus.

The access time is considered to last from t_1 to t_2. Then there is a recharge time, lasting from t_2 to t_3, during which DRAM chips will have to recharge before the processor can access them again.

a. Assume that the access time is 60 ns and the recharge time is 40 ns. What is the memory cycle time?

Memory cycle time = 60 + 40 = 100 ns.

b. What is the maximum data rate this DRAM can sustain, assuming a 1-bit output?

The maximum data rate is 1 bit every 100 ns, which is 10 Mb/s.

c. **Constructing a 32-bit wide memory system using these chips yields what data transfer rate in terms of Byte per second (B/s)?**

32×10 Mbps = 320 Mbps

= > 320 Mbps / 8 = 40 MB/s.

If one line of an assembler program (which loads contents of memory location 940 to the accumulator) for a hypothetical processor and corresponding machine code is given as:

address	assembler	machine code
300	LDA 940	1940

(the values are in hexd.)

a) **How many bits are needed for the program counter?**
Because the address size is 3 nibbles (3x4=12 bits), a 12-bit Program Counter is required.

b) **How many bits are needed for the instruction register?**
Instruction registers size equals to op-code size + address size (4+12), which is 16 bits.

c) **What is the maximum directly addressable memory capacity in KBytes?**
Maximum memory capacity = 2^{12} bytes = 2^{10+2} bytes = $2^2 \times 2^{10}$ bytes = 4 Kbytes

d) **For the memory location 940, find the Tag, Line, and Word values in hexadecimal format for a direct-mapped cache, when tag-id=4 bits, line-id=6 bits, word-id=2 bits.**
First, write the address in binary format:
(940)16 = (1001 0100 0000)2
Then define the Tag, Line, and Word values:
Tag-id = 1001 = 9
Line-id = 00010000 = 10
Word-id = 0000 = 0

Given the hexadecimal main memory address A1F85B73;

a) **How many bits are needed for the program counter?**
Because the address size is 4 bytes (32 bits), a 32-bit Program Counter is required.

b) **How many bits are needed for the instruction register if op-code is 8-bits.**
Instruction registers size equals to op-code size + address size, which is 40 bits

c) **What is the maximum directly addressable memory capacity in GBytes?**
Maximum memory capacity = 2^{32} bytes = 2^{30+2} bytes = $2^2 \times 2^{30}$ bytes = 4 GBytes

d) **Find the Tag, Line, and Word values in hexadecimal format for a direct-mapped cache, when *tag-id*=13 bits, *line-id*=16 bits, *word-id*=3 bits.**
First, write the address in binary format:
1010 0001 1111 1000 0101 1011 0111 0011

Tag-id = 0001010000111111 = 143F
Line-id = 0000101101101110 = 0B6E
Word-id = 0011 = 3

Consider a machine with a byte addressable main memory of 2^{16} bytes and block size of 8 bytes. Assume that a direct mapped cache consisting of 32 lines is used with this machine.

a. **How is the memory address divided into tag, line number, and byte number?**
word-id= $\log_2$(block size) = $\log_2$(8) = 3 bits
line-id = $\log_2$(#of lines) = $\log_2$(32) = 5 bits
tag-id = 16 – (5+3) = 8 bits

b. **Into what line would bytes with each of the following addresses be stored?**
I. 0001 0001 0001 1011
II. 1100 0011 0011 0100

I. 00010001 00011 011 → line id = 03
II. 11000011 00110 100 → line id = 06

X is a decimal number and its value is 51.

(a) **Represent X in unsigned binary integer form using 8 bits**
X = $(51)_{10}$ = $(00110011)_2$

(b) **Find the corresponding hexadecimal representation of X,**
X = $(51)_{10}$ = $(0011\ 0011)_2$ = $(33)_{16}$

(c) Assuming X is a 2s complement signed integer number using 8 bits, find –X.

$X = (51)_{10} = (00110011)_2$ ➔ $-X = -(51)_{10}$

2s complement of $(X)_2 = (11001101)_2$

(d) Represent X in BCD

$X = (51)_{10} = (0101\ 0001)_{BCD}$

Memory Location	Memory Content
000000	1010110010101100
000001	0010110110101100
.	.
111110	1010110010111101
111111	0000110110101100

The table shows a memory in a computer system.

(a) How many memory locations this memory has?

Because the address field of this memory is 6 bits, there are $2^6 = 64$ memory locations.

(b) What is the memory size (in Bytes)?

Each location stores 2 Byte (16 Bits) data. So, the memory size is number of locations × size of data at each location = 64×2 = 128 Byte

(c) Total how many bits this memory can store?

Because a byte is 8 bits, this memory can store maximum 128×8 = 1024 bits data

----

If one line of an assembler program for a hypothetical processor, which uses bytewise addressing, and corresponding machine code is given as:

(a) How many instructions this processor might have?

Because the op-code field of this instruction is 2 hexadecimal digits (8 bits) there might be maximum $2^8 = 256$ instructions

(b) How many bits are needed for the instruction register?

Because the op-code field of this instruction is 2 hexadecimal digits (8 bits) there might be maximum 28 = 256 instructions

(c) How many memories location an instruction will occupy?

Each location stores 1 Byte (8 Bits) data. The instruction length is 5 bytes. Therefore, each instruction will occupy 5 memory locations

(d) What is the total directly addressable memory size?

Address field of this instruction has 4 hexadecimal digits, so the total memory size is $2^{4\times4} = 2^{16}$ Byte $= 2^{10+6}$ Byte $= 2^6 \times 2^{10}$ Byte = 64 Kbyte.

Write a simple VVM Assembly program that adds the number entered from the keyboard and the data in memory location 20.

- If the result is positive, it is saved in memory location 30
- If the result is negative, it is saved in memory location 40

```
00      in      // input number
01      add 20 // accumulator + content of (20)
02      brp 05  // if + jump to 05
03      sto 40  // else store – number in (30)
04      jmp 06 // jump (branch) to 06
05      sto 30  // save the + number in (30)
06      hlt     // stop
```

In a computer system, address 100 contains decimal value 32, address 200 contains decimal value 10.

What would be the contents of accumulator after running the following assembler code?

LOAD 100
SHIFTR
SHIFTR
ADD 200

Instruction	Acc. Content	Operation
LOAD 100	A=32	$A \leftarrow M(100)$
SHIFTR	A=16	$A \leftarrow A/2$
SHIFTR	A=8	$A \leftarrow A/2$
ADD 200	A=18	$A \leftarrow A+M(200)$

List three broad classifications of external (or peripheral) devices.

- **Human readable**: Suitable for communicating with the computer user.
- **Machine readable**: Suitable for communicating with equipment.
- **Communication**: Suitable for communicating with remote devices

---

What is an instruction and instruction set?

Instruction: Language of the machine

Instruction set: Vocabulary of the language (collection of instructions that are understood by a CPU)

Consider a stack-based processor with instructions PUSH, POP, ADD, SUB, MUL, and DIV.

Write a program to compute X = (A + B2)/(D2 – E)

- PUSH A
- PUSH B
- PUSH B
- MUL
- ADD
- PUSH D
- PUSH D
- MUL
- PUSH E
- SUB
- DIV
- POP X

```
I0  LDA (50)
I1  ADD (60)
I2  BRN I7
I3  LDA (51)
………………
………………
I7  STA (70)
I8  JMP I3
```

A one-address type processor has an 8-bit accumulator (A), and a bytewise memory addressing. Data in memory locations 50 and 60 are 7A and 55 respectively, and they are in twos complement format. The following code is a part of an assembler program. If the code is executed;

a) **What will be the value of overflow flag after the execution of instruction I1?**
Overflow flag is 1.
Both 7A and 55 are positive. But the result is negative:

```
       7A     (01111010)
+      55     (01010101)
       CF     (11001111)
```

b) **What will be the value of carry flag after the execution of instruction I1?**
Carry flag will be 0, as there is no carry.

c) **Which line will be executed after the execution of instruction I2?**
Because the sign flag is 1 (the result is negative), instruction I7 will be executed after the execution of the instruction I2

d) **What will be the content of program counter (PC)after the execution of I8.**
The program counter content will be I3

Assume a stack-based processor that includes the stack operations PUSH and POP. Arithmetic operations automatically involve the top one or two stack elements. Begin with an empty stack.

What stack elements remain after the following instructions are executed?

Instruction	**Stack (top on the left)**
PUSH 4	4
PUSH 7	7, 4
PUSH 8	8, 7, 4
ADD	(8+7=15), 4
PUSH 10	10, 15, 4
SUB	(15-10=5), 4
MUL	(5×4=20)

Provide a typical list of the inputs and outputs of a control unit.

Inputs: Clock, Instruction register, Flags, Control signals from control bus:

Outputs: Control signals within the processor, Control signals to control bus:

In the following MIPS code.

```
lw     $t0,   32($s3)
add    $t0,   $s2,    $t0
sw     $t0,   48($s3)
```

a. **How many times is instruction memory accessed?**

   3

b. **How many times is data memory accessed?**

   2

c. **How many times is register file accessed?**

   7

d. **Which ones are read from the register file, which ones are written to the register file?**

lw $t0, 32($s3) ➔ s3 is read from,

t0 is written into

add $t0, $s2, $t0 ➔ s2 and t0 are read from,

t0 is written into

sw $t0, 48($s3) ➔ s3 and t0 are read from

Suppose a system has a virtual memory size of 64K and the system uses word addressing.

If the page size is 1024 Word,

Virtual address space: 64K = $2^6 \times 2^{10} = 2^{16}$ Word

Page size: 1024 Word = 2^{10} Word

a) **How many bits this virtual memory address requires?**

   Because the virtual memory size is 2^{16} Word, the virtual memory address requires: 16 bits

b) **Determine the number of virtual pages.**

   $2^{16} / 2^{10} = 2^{16-10} = 2^6 = 64$

Assume that you have a computer with a maximum main memory size of 2 GByte (motherboard does not allow to upgrade the main memory).

The program you want to run on this computer may require a larger main memory, resulting in a virtual memory that uses part of the hard drive.

However, you realized that the magnetic hard drive in this computer is too slow for your application.

What can you do to increase the performance of this computer? Explain how and why your solution will increase the performance.

I would replace magnetic hard drive with a Solid-State Drive (SSD).

SSDs have the following advantages over HDDs:

- High-performance input/output operations per second (IOPS),
- Less susceptible to physical shock and vibration
- Longer lifespan
- Lower power consumption
- Lower access times and latency rates
- Silent operation

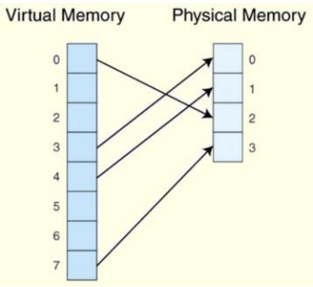
Suppose that a given process requires a virtual address space of 2^8 words and physical addresses in the computing system contain 7-bits. Assume also that pages are 32 words in length

a) **How many frames does the physical memory has?**

   $2^7 / 32 = 128 / 32 = 4$ frames

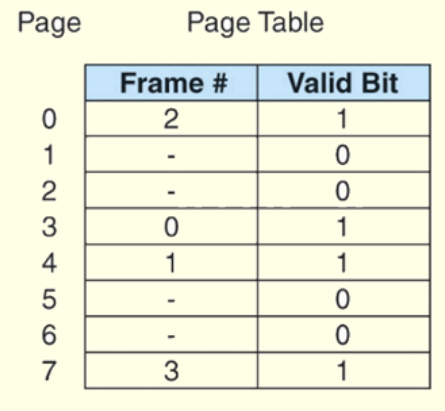
b) **How many pages does the virtual memory has?**

   $2^8 / 32 = 256 / 32 = 8$ pages



If some pages from the process have been brought into main memory as illustrated in the following figure,

c) What will be the contents of the page table?

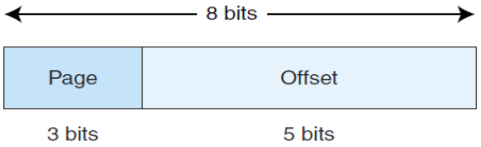


Page Page Table

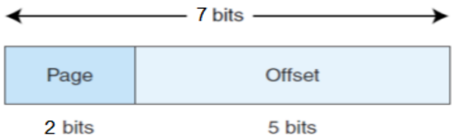
Page	Frame #	Valid Bit
0	2	1
1	-	0
2	-	0
3	0	1
4	1	1
5	-	0
6	-	0
7	3	1

The logical page number is translated into a physical page frame through a lookup in the page table.

d) What will be the format for the virtual address?



e) What will be the format for the physical address?



Suppose the system now generates the virtual address 13_{10} = $0D_{16}$ = 00001101_2.

f) What will be the values of page field and offset field of virtual address?

the page field = 0 = 000_2

the offset field = 13 = 01101_2

g) What will be the values of frame field and offset field of physical address?

the frame field = 2 = 10_2

the offset field = 13 = 01101_2

```
I1: ADD    R7, R5, R6
I2: MUL    R8, R7, R4
I3: ADD    R4, R2, R1
I4: DIV    R8, R1, R3
```

The code is a part of an assembler program written for a three-address machine type processor.

Identify the type of all possible dependencies in this code.

a) true data or write-read dependency:
I1, I2 (R7, R7)
I1: **ADD R7, R5, R6**
I2: **MUL R8, R7, R4**
I3: **ADD R4, R2, R1**
I4: **DIV R8, R1, R3**

b) output or write-write dependency:
I2, I4 (R8, R8)
I1: **ADD R7, R5, R6**
I2: **MUL R8, R7, R4**
I3: **ADD R4, R2, R1**
I4: **DIV R8, R1, R3**

c) anti-dependency or read-write dependency:
I2, I3 (R4, R4)
I1: **ADD R7, R5, R6**
I2: **MUL R8, R7, R4**
I3: **ADD R4, R2, R1**
I4: **DIV R8, R1, R3**

MIPS processor was born in the early 1980s from the work done by John Hennessy and his students at Stanford University and is mainly used in small devices. So,

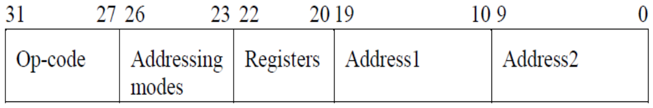
What architecture the MIPS processor adapted?

MIPS adapted the RISC architecture to explore the architectural concept of RISC

List the characteristics of the Von Neumann computer model.

- Three hardware systems:
  - A central processing unit (CPU)
  - A main memory system
  - An I/O system
- The capacity to carry out sequential instruction processing.
- A single data path between the CPU and main memory.

If a computer system, in which addressing is by 1 byte unit, has the following instruction format:



31 27	26 23	22 20	19 10	9 0
Op-code	Addressing modes	Registers	Address1	Address2

- **How many registers this computer system might have?**
  2^3 = 8 registers,
- **How many bits are needed for the program counter?**
  10 bits are needed
- **How many memory locations this instruction will occupy?**
  32 / 8 = 4 memory locations
- **Total how many bits can be stored in the memory of this computer?**
  2^{10} = 1024 Byte = 1024×8 = 8192 bit,

Consider the following assembly program written for a computer with a two-address instruction processor. If a portion of the memory is given as beside (V, W, X, Y, and Z are variables);

```
100   MOVE  R1, (200)
101   SUB   R1, (201)
102   MOVE  R2, (202)
103   ADD   R2, (204)
104   MPY   R1, R2
105   MOVE  (204), R1
```

Mem. Address	Data
200	V = A4
201	W = A0
202	X = 72
203	Y = 22
204	Z = 41

a) **What function (in terms of the given variables) this program implements? (R1 and R2 are registers; MOVE: data movement instruction; SUB: subtraction instruction; MPY: multiplication instruction; ADD: addition instruction; DIV: division instruction)**
Z = (V - W) (X + Y)

b) **Assuming that V, W, X, Y, and Z are 2s complement signed numbers, what will be the values of *carry flag*, *sign flag*, *zero flag*, and *overflow flag* after the execution of instruction in memory location 103? Explain**

```
              1
72 ➔          0 1 1 1 0 0 1 0
41 ➔          0 1 0 0 0 0 0 1
B3 ➔        0 1 0 1 1 0 0 1 1
```

carry flag = 0 , sign flag = 1,
zero flag = 0, overflow flag = 1

In an 8-bit microprocessor data transfer rate is given as 2 MBytes per second.

Find the clock frequency of the processor if a bus cycle takes 2 clock cycles.

- If 2 MBytes is transferred in 1 second, 1 Byte is transferred in $1/(2\times10^6) = 0.5\times10^{-6}$ second .
- If 1 byte requires 2 clock cycle to transfer, clock cycle is $(0.5\times10^{-6})/2 = 0.25\times10^{-6}$ second.
- So, the clock frequency is $1/(0.25\times10^6$ second) = 4 MHz

----

Classify and briefly describe I/O methods that can be used in a computing system.

- Programmed I/O: Reserves a register for each I/O device. Each register is continually polled to detect data arrival.
- Interrupt-Driven I/O: Allows the CPU to do other things until I/O is requested.
- Direct Memory Access (DMA):Offloads I/O processing to a special-purpose chip that takes care of the details.
- Channel I/O: Uses dedicated I/O processors.

A 16 bit microprocessor system has 10 address lines to address relevant memory locations.

a) **Assuming that the data size is 1 byte, what is the address of the last memory location?**
The last memory location is
$= 2^{\#\text{of bits in address line}} - 1 = 2^{10} - 1 = 1023$

b) **Assuming memory chips organized as 512x8 bits are used, how many memory chips we need to design the required memory).**
Data bus size **16 bits**, memory size is **1024**. System memory organization is **1024x16**.

c) **Design the required memory system using the memory chips organized as 512x8 bits.**
If we restate this memory organization in terms of the available memory chip (512x8), we have {(2x512) x (2x8)} = **4**x(512x8).
So, we need **4** (512x8) memory chips to have a memory organized as 1024x16

Why a cash and virtual memory are used in a computing system? Briefly explain.

Cache memory enhances performance by providing faster memory access speed.

- The purpose of cache memory to speed up accesses by storing recently used data closer to the CPU.
- It is much smaller than main memory. Its access time is a fraction of that of main memory.

Virtual memory enhances performance by providing greater memory capacity, without the expense of adding main memory.

- Instead, a portion of a disk drive serves as an extension of main memory.

Suppose we have 32-bit memory addresses, a byte-addressable memory, and a 512 KB cache with 32 bytes per block.

1. **How many total lines are in the cache?**

(512 KBytes)/(32 Bytes)= 2^19/2^5 =2^14 lines
=> 2^4×2^10 lines=16 K lines=16384 lines

2. **If the cache is direct-mapped, how many cache lines could a specific memory block be mapped to?**
1 (a specific memory block could be mapped to only 1 line in the cache)

3. **If the cache is direct-mapped, what would be the format (Tag-id, Line-id, Word-id) of the address? (Clearly indicate the number of bits in each)**
Each block contains 32 bytes. Therefore, 5 bits are needed to specify the *Word-id*.

There are 2^{14} lines in the cache. Therefore, 14 bits are needed to specify the *Line-id*.

Tag-id bits in this case is defined as *Main memory address* – (*Line-id bits* + *Word-id bits*). Therefore, 32 - (14 + 5) = 13 bits are needed to specify the *Tag-id*.

The format of the address:

Tag-id	Line-id	Word-id
13 bits	14 bits	5 bits

4. **If the cache is 2-way set associative (sets of size 2), how many cache lines could a specific memory block be mapped to?**

2 (a specific memory block could be mapped to any 2 lines within 1 set in the cache)

5. **If the cache is 2-way set associative (sets of size 2), how many sets would there be?**

(2^14 lines) / (2^1 lines) = 2^13 sets
=> 2^3×2^10 sets=8 K sets=80192 sets

6. **If the cache is 2-way set associative, what would be the format of the address?**
Each block contains 32 bytes. Therefore, 5 bits are needed to specify the *Word-id*.

There are 2^{13} sets in the cache. Therefore, 13 bits are needed to specify the *Set-id*.

Tag-id bits in this case is defined as *Main memory address* – (*Set-id bits* + *Word-id bits*). Therefore, 32 - (13 + 5) = 14 bits are needed to specify the *Tag-id*.

Tag-id	Set-id	Word-id
14 bits	13 bits	5 bits

7. **If the cache is fully-associative, how many cache lines could a specific memory block be mapped to?**
   2^{14} (a specific memory block could be mapped to any cash line in the cache)

8. **If the cache is fully-associative, what would be the format of the address?**
   Each block contains 32 bytes. Therefore, 5 bits are needed to specify the *Word-id*.

   Tag-id bits in this case is defined as *Main memory address – Word-id bits*. Therefore, 32 - 5 = 27 bits are needed to specify the *Tag-id*.

Tag-id	Word-id
27 bits	5 bits